1 NOV 2000

AFRL-VA-WP-TR-2000-3045

DEVELOPMENT OF THE AERODEYNAMIC/AEROELASTIC MODULES IN ASTROS

VOLUME I - AEROSERVOELASTICITY DISCIPLINE IN ASTROS USER'S MANUAL

ZONA TECHNOLOGY INC
7434 E STETSON DR SUITE 205
SCOTTSDALE AZ 85251

FINAL REPORT FOR 9/1/96 -- 9/30/98

Approved for public release; distribution unlimited.

Air Vehicles Directorate
Air Force Research Laboratory
Air Force Materiel Command
Wright-Patterson AFB, OH 45433-7542

NOTICE

USING GOVERNMENT DRAWINGS, SPECIFICATIONS, OR OTHER DATA INCLUDED IN THIS DOCUMENT FOR ANY PURPOSE OTHER THAN GOVERNMENT PROCUREMENT DOES NOT IN ANY WAY OBLIGATE THE UNITED STATES GOVERNMENT. THE FACT THAT THE GOVERNMENT FORMULATED OR SUPPLIED THE DRAWINGS, SPECIFICATIONS, OR OTHER DATA DOES NOT LICENSE THE HOLDER OR ANY OTHER PERSON OR CORPORATION; OR CONVEY ANY RIGHTS OR PERMISSION TO MANUFACTURE, USE, OR SELL ANY PATENTED INVENTION THAT MAY BR RELATED TO THEM.

THIS REPORT IS RELEASEABLE TO THE NATIONAL TECHNICAL INFORMATION SERVICE (NTIS). AT NTIS, IT WILL BE AVAILABLE TO THE GENERAL PUBLIC, INCLUDING FOREIGN NATIONS.

THIS TECHNICAL REPORT HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION.

Victoria A. Tischler
VICTORIA A. TISCHLER
Aerospace Engineer
Structural Design & Development Branch

Nelson D. Wolf
NELSON D. WOLF, Chief
Structural Design and Development Branch
Structures Division

J. M. Manter
JOSEPH M. MANTER, Chief
Structures Division
Air Vehicles Directorate

COPIES OF THIS REPORT SHOULD NOT BE RETURNED UNLESS RETURN IS REQUIRED BY SECURITY CONSIDERATIONS, CONTRACTUAL OBLIGATIONS, OR NOTICE ON A SPECIFIED DOCUMENT.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
FEBRUARY 04, 1999

3. REPORT TYPE AND DATES COVERED
FINAL 24 SEP 1996 - SEP 1998

4. TITLE AND SUBTITLE
DEVELOPMENT OF THE AERODYNAMIC/AEROSERVOELASTIC MODULES IN ASTROS / VOLUME I - AEROSERVOELASTICITY DISCIPLINE IN ASTROS USER'S MANUAL

5. FUNDING NUMBERS
C: F33615-96-C-3217
PE: 65502F
PR: STTR
TA: 41
WU: 00

6. AUTHOR(S)
M. Karpel and B. Moulin
Technion - I.I.T.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Subcontractor to Research Insitutute:
Technion - Israel Institute of Technology
Haifa 32000
Israel
Tel 972-4-8293490 / Fax 972-4-8229352

Prime Contractor:
ZONA Technology, Inc.
7434 E. Stetson Drive, Suite 205
Scottsdale, AZ 85251
Tel (602) 945-9988 / Fax (602) 945-6588

8. PERFORMING ORGANIZATION REPORT NUMBER
ZONA 99-11E

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Vehicles Directorate
Air Force Research Laboratory
Air Force Materiel Command
Wright Patterson Air Force Base OH 45433-7531
POC: Capt. Gerald Andersen, (937) 255-6992 / Dr. V.B.Venkayya (937) 255-2582

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-VA-WP-TR-2000-3045

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The behavior of the structural and control systems of flight vehicles are highly coupled through aeroelastic effects. An aeroservoelastic (ASE) interaction module was developed to facilitate ASE analysis and the application of ASE stability and response constraints within ASTROS. The aeroelastic plant state-space equations are based on minimum-state rational function approximation of the unsteady force coefficient matrices. The control system is defined in a way that allows incorporation of most general linear control laws in the aeroservoelastic loop, and yet allows efficient control margin and sensitivity computations by separating between changeable gains to other control elements and parameters. The new analysis and sensitivity items include open- and closed-loop flutter, control SISO gain and phase margins, singular values, aeroelastic system gains, and continuous gust response. This report is part of the documentation which describe the complete development of an STTR Phase II effort entitled "Development of the Aerodynamics/Aeroservoelasticity Modules in ASTROS". Additional aeroservoelasticity (ASE) reports are the Theoretical Manual, the Programmer's Manual, and the Application Manual.

14. SUBJECT TERMS
Multidisciplinary Optimization, Aeroelasticity, Aeroservoelasticity, Gust Response, ASTROS

15. NUMBER OF PAGES
60

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAR

Standard Form 298 (Rev. 2-89) (EG)
Prescribed by ANSI Std. 239.18
Designed using Perform Pro, WHS/DIOR, Oct 94

FOREWORD

This final report is submitted in fulfillment of CDRL CLIN 0001, Data Item A001, Title: Scientific and Technical Reports of a Small business Technology Transfer (STTR) Phase II contract No. F33615-96-C-3217 entitled, "Development of the Aerodynamic/Aeroservoelastic Modules in ASTROS," covering the performance period from 24 September 1996 to 24 September 1998.

This work is the second phase of a continuing two-phase STTR contract supported by AFRL/Wright-Patterson. The first phase STTR contract No. F33615-95-C-3219 entitled, "Enhancement of the Aeroservoelastic Capability in ASTROS," was completed in May 1996 and published as WL-TR-96-3119.

Both STTR Phase I and Phase II contracts are performed by the same ZONA Team in which ZONA Technology, Inc. is the prime contractor, whereby the team members include: the University of Oklahoma (OU), Universal Analytics, Inc. (UAI), and Technion (I.T.T.).

This final report consists of eight volumes, these are:

*ASTROS**

Volume I	-	ZAERO User's Manual
Volume II	-	ZAERO Programmer's Manual
Volume III	-	ZAERO Application Manual
Volume IV	-	ZAERO Theoretical Manual

ASTROServo

Volume I	-	Aeroservoelastic Discipline in ASTROS, User's Manual
Volume II	-	Aeroservoelastic Discipline in ASTROS, Programmer's Manual
Volume III	-	Aeroservoelastic Discipline in ASTROS, Application Manual
Volume IV	-	Aeroservoelastic Discipline in ASTROS, Theoretical Manual

This document (Volume I) is the User's Manual of the Aeroservoelastic (ASE) interaction module developed to facilitate ASE analysis and the application of ASE stability and response constraints within ASTROS.

At AFRL/Wright-Patterson, Captain Gerald Andersen was the contract monitor and Dr. V. B. Venkayya was the initiator of the whole STTR effort. The technical advice and assistance received from Mr. Doug Niell of the MacNeal Schwendler Corporation, Dr. V. B. Venkayya and others from AFRL during the course of the present phase on the development of ASTROS* are gratefully acknowledged.

Contents

Chapter 1

Introduction

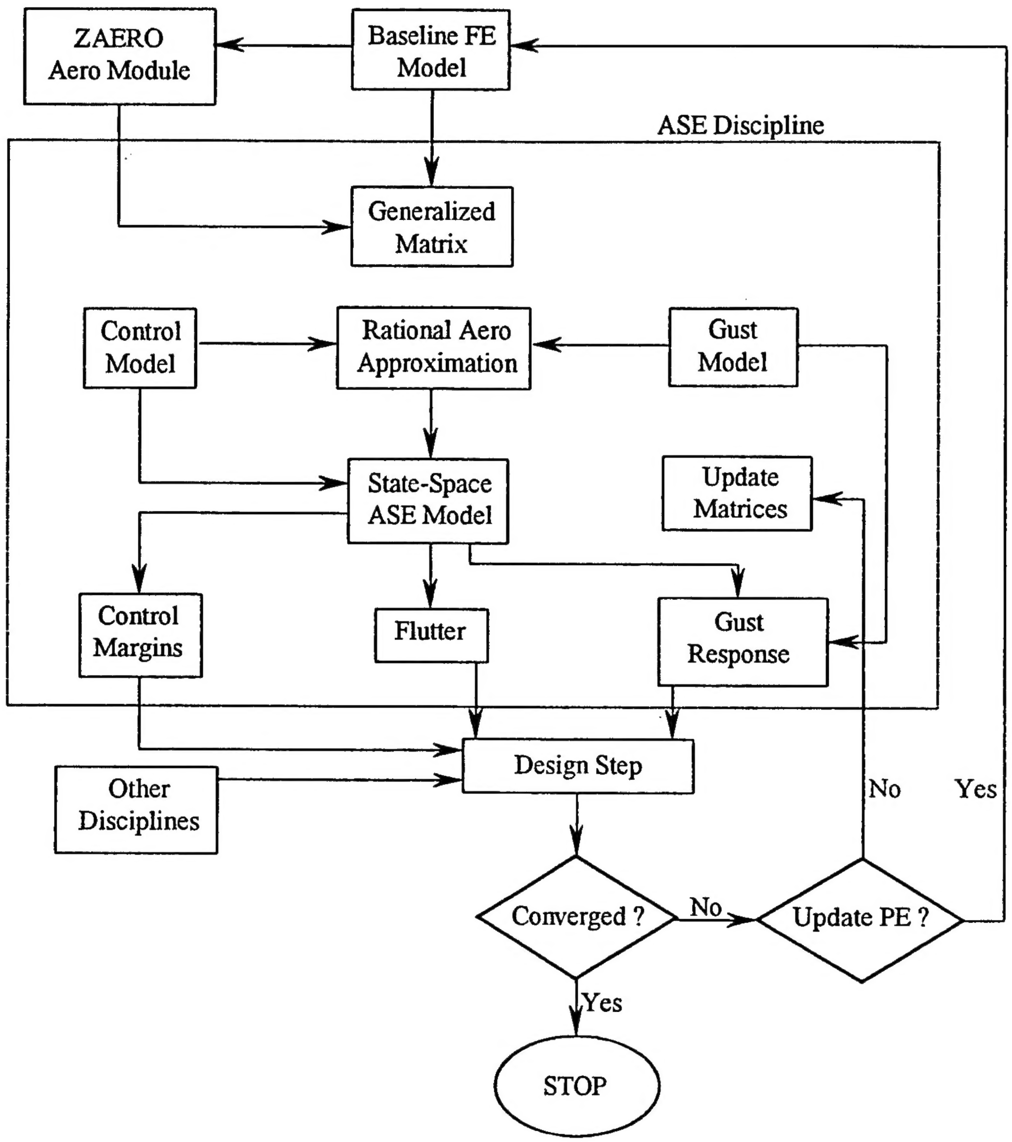
The User's Manual describes the User's interface with Version 11 of ASTROS for using the new aeroservoelastic (ASE) discipline. The new software is being developed under the US Air Force STTR Phase II contract (Topic No. AF95T009). The product of Phase II is in the form of new engineering application modules and data-input templates that are integrated into ASTROS by MAPOL command sequences that alter the standard MAPOL sequence. The MAPOL sequence changes are made in two steps:
1) The changes associated with the new ZAERO aerodynamic module are introduced first, and ASTROS is compiled. The new compiled version is called ASTROZ.
2) The ASE changes are introduced in alter commands that refer to the ASTROZ MAPOL sequence.
The ASE alterations are in both the optimization and the final analysis portions of the standard MAPOL sequence. A general flow chart of the new ASE discipline as part of a multidisciplinary optimization process is given in Figure 1.1.

It is assumed that the user is familiar with version 11 of ASTROS and its manuals. We try to keep here the same style and formats as in the ASTROS manuals such that the ASE manuals can be easily integrated in the general ones.

The new discipline is based on state-space formulation and it facilitates the fixed-basis modal approach in which entire optimization runs (or major parts of them) are performed without updating the baseline finite-element model. The fixed basis is based on a set of low-frequency normal modes of a baseline structure. The new approach leads to some major differences in the way generalized structural and aerodynamic matrices are stored in the data base. There are also significant differences between the analysis of a baseline structure and those of subsequent design iterations.

To appreciate the advantages of the fixed-basis modal approach, one has to realize that design optimization is an interactive process in which numerous optimization trials are performed with various design cases, constraint values, move limits, side limits and approximation levels. The fixed-basis approach facilitates a design scenario in which a computationally heavy run is first performed to create a modal data base. The analyst can then inspect the behavior of the baseline model and then perform numerous optimization runs in a very efficient on-line manner.

The various ASE run options and the associated MAPOL sequences are described in Chapter 2. Solution control and output features of the ASE module are discussed in Chapter 3. The new bulk data templates are described in Chapter 4. The new engineering application modules are described in the Programmer's Manual.


ZAERO Aero Module
Baseline FE Model
ASE Discipline
Generalized Matrix
Control Model
Rational Aero Approximation
Gust Model
State-Space ASE Model
Update Matrices
Control Margins
Flutter
Gust Response
Design Step
Other Disciplines
No
Yes
Converged ?
No
Update PE ?
Yes
STOP

Chapter 2

MAPOL Sequences

2.1 Run options

The ASE discipline facilitates an efficient design process with a data base created in previous runs. Hence, we can categorize the run options according to whether or not a data base is created. The options are controlled by the ASEDB parameter of the ASEUP bulk-data entry. The optional runs are:

1. Baseline analysis: an analysis run which creates a data base for subsequent analysis and optimization runs (ANALYZE with ASEDB = 1).

2. Analysis with optimal modifications: a restart run which may modify the baseline modal matrices according to user-input global design variables, and performs analysis in baseline coordinates (ANALYZE with ASEDB = 2).

3. Optimization by restart: a restart run that uses the modal and sensitivity data base created in Option 1. This option is designed for cases were the ASE discipline is treated in the entire optimization with a fixed basis. The optimization is followed by a final analysis with an updated model (OPTIMIZE with ASEDB = 2).

4. Cold-start optimization: the baseline analysis, the subsequent optimization and the final analysis are performed in the same run. A new modal ASE data base, based on an updated finite-element model, is created every $\mathbf{n}_{up}$ iterations (ASEDB = 0).

2.2 MAPOL changes

2.2.1 Description of the changes

The sequence of MAPOL changes for integrating the ASE discipline in ASTROS Version 11, after adding the ZAERO aerodynamics (ASTROZ), is given in section 2.3.2. The main portions of the current MAPOL changes are:

1. Insert 241 in the preface part for declaration of the new variables and definition of the new procedures.

2. Insert 326 in the preface part for defining the run options of the ASE discipline run. The new ASERUN module is applied here to extract data from the new relation ASEUP and create three logical flags: ASEDIS which is set to TRUE if ASE discipline is employed; ASEDBMAK which is set to TRUE if modal data base is created in analysis run; ASEOPTDB which is set to TRUE if modal data base is created in the first iteration of optimization run (cold start). Then unsteady aerodynamics modules are skipped if ASEDIS is TRUE and both ASEDBMAK and ASEOPTDB are FALSE.

3. Insert 406 to make logical flag BCFASE to the current boundary conditions using the new module ASEBC.

4. Insert 383 in which the new UPDV module is applied to extract the modified design values from the new relation VINIG and create the logical flag DVFLG which is set to TRUE if an update is performed (relation VINIG is not empty).

5. Inserts 450, 581, 585, 635, 639, and 661 to skip reduction of the structural matrices if BCFASE is TRUE and ASEOPTDB is FALSE.

6. Inserts 677, 683, 684, 693, 699, and 700 to apply regular normal modes analysis and build generalized stiffness and mass matrices of the baseline using the new module GMAT if ASEOPTDB is TRUE.

7. Insert 702 to modify generalized stiffness and mass matrices by the new GMGKUPD module, perform a reduced basis eigenvalue analysis by the new RMODE module and recover modes to the g-set by the MAPOL procedure AGRECOV.

8. Inserts 899, 907, and 911 modify the flutter discipline. The new module ASEDRV creates a logical flag LASE for the current subcase which is set to TRUE if ASE flutter discipline is to be employed. If LASE is FALSE standard flutter analysis is performed using modules FLUTQHHZ, FLUTDMA, FLUTTRAZ. If ASEOPTDB is TRUE the standard FLUTQHHZ and FLUTDMA modules as well as the new BDMAT and MIST modules are applied to build dynamic and aerodynamic approximation matrices that form the baseline for the updated modal data base. If ASEOPTDB is FALSE the new module MGFL is used to modify dynamic, aerodynamic approximation matrices and sensitivities of generalized

matrices. Independently to ASEOPTDB (if LASE is TRUE) the new module ASEFLUT replaces the standard flutter analysis module FLUTTRAZ.

9. Inserts from 976 to 1243 to avoid recovery of the modes if ASE discipline is employed for the current boundary conditions.

9. Inserts 1466, 1467, 1474, 1475, 1482 modify the flutter sensitivity segment. The new module ASEDRV again creates a logical flag LASE for the current subcase. If LASE is set to FALSE standard flutter sensitivity procedure is performed using module FLUTSENZ; otherwise it is replaced by the new module ASESENS. If ASEOPTDB is TRUE sensitivities of generalized matrices are built before the module ASESENS.

10. Insert 1485 in which updating of the logical flag ASEOPTDB is performed according to the user-defined parameter NUP.

10.Insert 1972 is the first one that affects on the final analysis segment. Here the new UPDV module is applied to extract the modified design values from the new relation VINIG and create the logical flag DVFLG which is set to TRUE if the updating was done (relation VINIG was not empty). Note that these operations are done only if ASEDIS Is TRUE and if optimization was not performed before analysis in the current run. Otherwise analysis is performed for the design variables values obtained in the optimization process (where VINIG values were already employed).

11.Inserts 1990, 2144, 2148, 2198, 2202, and 2217 to skip reduction of the structural matrices if DVFLG is TRUE.

12. Inserts 2232, 2238, 2247, 2253, and 2254 to apply regular normal modes analysis and build generalized stiffness and mass matrices of the baseline using the new module GMAT if DVFLG is FALSE or to modify generalized stiffness and mass matrices by the new GMGKUPD module and perform a reduced basis eigenvalue analysis by the new RMODE module if DVFLG is TRUE.

13. Inserts 2440, 2448 and 2451 modify the flutter discipline. The new module ASEDRV creates a logical flag LASE for the current subcase which is set to TRUE if ASE flutter discipline is to be employed. If LASE is FALSE standard flutter analysis is performed using modules FLUTQHHZ, FLUTDMA, FLUTTRAZ. If DVFLG is FALSE the standard FLUTQHHZ and FLUTDMA modules as well as the new BDMAT and MIST modules are applied to build dynamic and aerodynamic approximation matrices. If DVFLG is TRUE the new module MGFL is used to modify the dynamic and aerodynamic approximation matrices. Independently to ASEOPTDB (if LASE is TRUE) the new ASEFLUT module replaces the standard flutter analysis module FLUTTRAZ.

14. Insert 2741 for creating sensitivities of generalized stiffness and mass matrices and for flashing the old ASEUP entity if ASEDIS and ASEDBMAK are TRUE.

2.2.2 MAPOL changes

```
   EDIT NOLIST
REPLACE 193
      [AJC],    [SCNTLK],   [ACNTLK],
INSERT 241
$***$
$     *************   FILE "ase_un.mpl"   *************      $
$                                             $
$            AEROSERVOELSTIC DISCIPLINE                  $
$             WITH ZONA AERODYNAMICS                     $
$                                             $
$   Boris Moulin and Moti Karpel                       $
$   Technion - Israel Institute of Technology           $
$   Faculty of Aerospace Enginearing                   $
$   December, 1998                                $
$***$
$************************************************************************
*$
$      VARIABLE DECLARATION SEGMENT                       $
$************************************************************************
*$
INTEGER   IPA;
INTEGER   NUP,      NITERUP,   ISYM;
LOGICAL   DVFLG,     ASEDIS,     ASEDBMAK,   ASEOPTDB,   LASE;
LOGICAL   USAMDL,     CONTRFL,    BCFASE;
RELATION  ASESOL,    ASEUP;
RELATION  VINIG,     ASELAMBD,   ASEMARG;
RELATION  MINSTAT,    AEROLAG,    AEROGND,    PWEIGHT,    DINIT;
RELATION  APCONST,    APCNSND;
RELATION  ASECONT,    SISOTF,     MIMOSS;
RELATION  ASESNSR,    ACTU,      CJUNCT,     CONCT,      ASEGAIN;
RELATION  GAINSET,    CNCTSET,    SURFSET,    SENSET,     TFSET;
RELATION  CMARGIN,    DCONUGM,    DCONLGM,    DCONUPM,    DCONLPM;
RELATION  GAINMPC,    DCONGAIN;
RELATION  CONGUST,    RESPSET,    CRESP;
MATRIX    [DTMSB(30,33)],      [EMSB(30,33)],       [AMSB(30,33)];
MATRIX    [DTMS(30,33)],       [EMS(30,33)],        [AMS(30,33)];
MATRIX    [RMS(30,33)],        [PHFLO(30,33)],      [KRHHFL(30,33)];
MATRIX    [DGKV(1000)],        [DGMV(1000)];
MATRIX    [DGKVB(1000)],       [DGMVB(1000)];
MATRIX    [DGMVC(30,33)],      [DGMCB(30,33)];
MATRIX    [GKB(1000)],         [GMB(1000)];
MATRIX    [GK(1000)],          [GM(1000)],          [PHIAB(1000)];
```

```
MATRIX   [GM2(30,33)],        [GB2(30,33)],        [GK2(30,33)];
MATRIX   [NLCR(30,33)],       [PLROG(30,33)];
MATRIX   [QHC(30,33)],        [MCC(30,33)],        [MHC(30,33)];
MATRIX   [MIC(30,33)],        [MICB(30,33)];
MATRIX   [QHG(30,33)];
MATRIX   [PSI],    [DELGM],    [DELGK],    [DELDV];
MATRIX   [ACNTLG],    [SCNTLG];
$                                   $
$*************************************************************************
*$
$       PROCEDURES                              $
$*************************************************************************
*$
$                                   $
$ PROCEDURE FOR REDUCTION FROM G-SET TO A-SET                         $
$                                   $
PROC GAREDUCE ( [MATRG], [MATRN], [MATRS], [MATRF], [MATRA], [MATRO] );
MATRIX   [MATRG],    [MATRN],    [MATRS],    [MATRF],    [MATRA];
MATRIX   [MATRO];
  IF NMPC <> 0 THEN
     CALL GREDUCE ( , [MATRG], [PGMN(BC)], [TMN(BC)], , [MATRN] );
  ELSE
    [MATRN] := (1) [MATRG];
  ENDIF;
  IF NSPC <> 0 THEN
    CALL NREDUCE ( , [MATRN], [PNSF(BC)], , , , , [MATRF], [MATRS] );
  ELSE
    [MATRF] := (1) [MATRN];
  ENDIF;
  IF NGDR <> 0 THEN
    [MATRA]    := TRANS ( [GSUBO(BC)] ) * [MATRF];
  ELSE
    IF NOMIT <> 0 THEN
      CALL ROWPART ( [MATRF], [MATRO], [UGTKAB], [PFOA(BC)] );
      [TMP1]  := TRANS( [MATRO] ) * [GSUBO(BC)];
      CALL TRNSPOSE ( [TMP1], [TMP2] );
      [MATRA] := [UGTKAB] + [TMP2];
    ENDIF;
  ELSE
    [MATRA] := (1) [MATRF];
  ENDIF;
ENDP;
$                                   $
$                                   $
$ PROCEDURE FOR RECOVERY MODES FROM A-SET TO G-SET                         $
```

```
$                                                    $
PROC AGRECOV ( [MATRA], [MATRO], [MATRF], [MATRN], [MATRM], [MATRG]);
MATRIX    [MATRA],    [MATRG],    [MATRN],    [MATRF],    [MATRM];
MATRIX    [MATRO];
  IF NGDR <> 0 THEN
    [UFGDR]   := [GSUBO(BC)] * [MATRA];
    CALL ROWPART ( [MATRA], [UJK], , [PAJK] );
    CALL ROWMERGE ( [MATRF], [UJK], [UFGDR], [PFJK] );
  ELSE
    IF NOMIT <> 0 THEN
      [MATRO] := [GSUBO(BC)] * [MATRA];
      CALL ROWMERGE ( [MATRF], [MATRO], [MATRA], [PFOA(BC)] );
    ELSE
      [MATRF]   := (1)[MATRA];
    ENDIF;
  ENDIF;
  IF NSPC <> 0 THEN
    CALL YSMERGE ( [MATRN], [YS(BC)], [MATRF], [PNSF(BC)] );
  ELSE
    [MATRN] := (1)[MATRF];
  ENDIF;
  IF NMPC <> 0 THEN
    [MATRM] := [TMN(BC)] * [MATRN];
    CALL ROWMERGE ( [MATRG], [MATRM], [MATRN], [PGMN(BC)] );
  ELSE
     [MATRG] := (1)[MATRN];
  ENDIF;
ENDP;
$                                                    $
$                                                    $
$ PROCEDURE TO MAKE SENSITIVITIES OF CONTROL MODES MASS COUPLING
$
$                                                    $
PROC CNTSENS ( [MATRF], [MATRG], [DGMVCB] );
MATRIX    [MATRG],    [MATRF],    [DGMVCB];
$                                                    $
  IF NSPC <> 0 THEN
    CALL YSMERGE ( [TMP1], [YS(BC)], [MATRF], [PNSF(BC)] );
  ELSE
    [TMP1] := (1)[MATRF];
  ENDIF;
  IF NMPC <> 0 THEN
    [TMP2] := [TMN(BC)] * [TMP1];
    CALL ROWMERGE ( [MATRG], [TMP2], [TMP1], [PGMN(BC)] );
  ELSE
```

```
    [MATRG] := (1)[TMP1];
  ENDIF;
  CALL MAKDVU ( 1, NDV, GLBDES, [MATRG], [DMAG], GMMCT, DMVI );
  [DGMVCB] := -TRANS([PHIG(BC)]) * [DMAG];
ENDP;
$                                                        $
INSERT 326
$                                                        $
$  ASERUN MAKE FLAGS:                                                 $
$    ASEDIS = TRUE, IF ASE DISCIPLINE IS EMPLOYED (CARD ASEUP IS IN THE
BULK) $
$    ASEDBMAK = TRUE, IF MODAL DATA BASE IS CREATED IN ANALYSIS
$
$    ASEOPTDB = TRUE, IF MODAL DATA BASE IS CREATED IN OPTIMIZ. (COLD
START)  $
$                                                        $
CALL ASERUN ( ASEDIS, ASEDBMAK, ASEOPTDB, NUP );
NITERUP := 1;
IF NOT ASEDIS OR ASEDBMAK OR ASEOPTDB THEN
INSERT 369
ENDIF;
INSERT 383
$                                                        $
$  FIND VINIG CARDS IN THE BULK DATA AND UPDATE DESIGN VARIABLES
$
$                                                        $
  CALL DFREL ( ASELAMBD, 2 );
  CALL UPDV ( VINIG, GLBDES, NDV, DVFLG, [DELDV] );
INSERT 406
$                                                        $
$       MAKE ASEBC FLAG FOR THE CURRENT BOUNDARY CONDITION                $
$                                                        $
      CALL ASEBC ( BC, BCFASE);
INSERT 450
IF NOT BCFASE OR ASEOPTDB THEN
INSERT 581
ENDIF;
INSERT 585
IF NOT BCFASE OR ASEOPTDB THEN
INSERT 635
ENDIF;
INSERT 639
IF NOT BCFASE OR ASEOPTDB THEN
INSERT 661
ENDIF;
```

```
INSERT 677
IF NOT BCFASE OR ASEOPTDB THEN
INSERT 683
$                                                   $
$            CREATE GENERALIZED MATRICES OF THE BASELINE              $
$                                                   $
           CALL GMAT ( NITER, BC, LAMBDA, [MII], [GMB(BC)], [GKB(BC)] );
           [PHIAB(BC)] := (1) [PHIA];
           CALL AGRECOV ( [PHIA], [PHIO], [PHIF], [PHIN], [UM], [TMP1]);
           [PHIG(BC)] := (1) [TMP1];
INSERT 684
ENDIF;
INSERT 693
IF NOT BCFASE OR ASEOPTDB THEN
INSERT 699
$                                                   $
$            CREATE GENERALIZED MATRICES OF THE BASELINE              $
$                                                   $
           CALL GMAT ( NITER, BC, LAMBDA, [MII], [GMB(BC)], [GKB(BC)] );
           [PHIAB(BC)] := (1) [PHIA];
           CALL AGRECOV ( [PHIA], [PHIO], [PHIF], [PHIN], [UM], [TMP1]);
           [PHIG(BC)] := (1) [TMP1];
INSERT 700
ENDIF;
INSERT 702
      IF BCFASE AND NOT ASEOPTDB THEN
        IF BMODES <> 0 THEN
          IF NITER <> 1 OR DVFLG THEN
$                                                   $
$            CREATE GENERALIZED MATRICES OF THE MODIFIED STRUCTURE          $
$                                                   $
           CALL GMGKUPD ( NITER, NITERUP, NDV, DVFLG,
                   [DGMV(BC)], [DGKV(BC)],
                   [DELDV], GLBDES, [GMB(BC)], [GKB(BC)],
                   [DELGM], [DELGK], [GM(BC)], [GK(BC)],
                   [DGMVC(BC,SUB)], [MICB(BC,SUB)],
                   [MIC(BC,SUB)] );
$                                                   $
$            REDUCED MODAL ANALYSIS OF THE MODIFIED STRUCTURE          $
$                                                   $
           CALL RMODE ( NITER, BC, HSIZE(BC), USET(BC), [GK(BC)],
                   [GM(BC)], , ,LAMBDA, [PSI], [MII], IPA );
           CALL OFPMROOT ( NITER, BC, NUMOPTBC, LAMBDA );
           CALL FCEVAL ( NITER, BC, LAMBDA, CONST );
           [PHIA] := [PHIAB(BC)] * [PSI];
```

```
          CALL AGRECOV ( [PHIA], [PHIO], [PHIF], [PHIN], [UM], [TMP1]);
          [PHIG(BC)] := (1) [TMP1];
        ELSE
$                                                    $
$          BASELINE STRUCTURE                                    $
$                                                    $
          CALL BASLUPD ( NITER, LAMBDA );
$           CALL OFPMROOT ( NITER, BC, NUMOPTBC, LAMBDA );$
          CALL FCEVAL ( NITER, BC, LAMBDA, CONST );
        ENDIF;
      ENDIF;
    ENDIF;
INSERT 899
$                                                    $
$          ASEDRV MAKE FLAG: LASE = TRUE, IF METHOD = ASE              $
$                                                    $
          CALL ASEDRV ( BC, SUB, LASE );
          LASE := TRUE;
          IF NOT LASE OR ASEOPTDB THEN
INSERT 907
            IF ASEOPTDB  CALL BDMAT ( NITER, BC, SUB, HSIZE(BC),
                           ESIZE(BC), [GMB(BC)], [GKB(BC)],
                           [MHHFL(BC,SUB)], [KHHFL(BC,SUB)],
                           [BHHFL(BC,SUB)], [KRHHFL(BC,SUB)],
                           [GM2(BC,SUB)],[GK2(BC,SUB)],
                           [GB2(BC,SUB)] );
          ELSE
$                                                    $
$           UPDATE MATRICES FOR FLUTTER                          $
$                                                    $
            IF NITER <> 1 OR DVFLG THEN
              CALL MGFL ( NITER, BC, SUB, HSIZE(BC), ESIZE(BC),
                      LAMBDA, [MII], [PSI], [GM2(BC,SUB)],
                      [GK2(BC,SUB)], [GB2(BC,SUB)],
                      [MHHFL(BC,SUB)], [KRHHFL(BC,SUB)],
                      [BHHFL(BC,SUB)], [DTMSB(BC,SUB)],
                      [EMSB(BC,SUB)], [AMSB(BC,SUB)],
                      [DTMS(BC,SUB)], [EMS(BC,SUB)],
                      [AMS(BC,SUB)],
                      [MIC(BC,SUB)], [MHC(BC,SUB)],
                      [DGMVB(BC)], [DGKVB(BC)], [DGMCB(BC,SUB)],
                      [DGMV(BC)], [DGKV(BC)], [DGMVC(BC,SUB)] );
            ENDIF;
          ENDIF;
          IF NOT LASE THEN
```

```
INSERT 911
          ELSE
            IF ASEOPTDB THEN
$                                                        $
$              MAKE CONTROL AND GUST MODES                          $
$                                                        $
              CALL CNTMOD ( NITER, BC, SUB, [AJC], [SKJ], [SCNTLK],
                        [ACNTLK], [SCNTLG], [ACNTLG], REUNMK,
                        [PHIKH], [PHIG(BC)], [MGG], [QHC(BC,SUB)],
                        [MCC(BC,SUB)], [MICB(BC,SUB)],
                        CONTRFL, ISYM );
              [MIC(BC,SUB)] := (1) [MICB(BC,SUB)];
              [MHC(BC,SUB)] := (1) [MICB(BC,SUB)];
              CALL GSTMOD ( NITER, BC, SUB, [QGK], REUNMK, [PHIKH],
                        [QHG(BC,SUB)] );
$                                                        $
$              MAKE SENSITIVITIES FOR CONTROL MODES MASS COUPLING    $
$                                                        $
              IF CONTRFL THEN
                IF ISYM = 1 THEN
                  CALL MAKDVU ( NITER, NDV, GLBDES, [SCNTLG],
                          [DMAG], GMMCT, DMVI );
                ELSE
                  CALL MAKDVU ( NITER, NDV, GLBDES, [ACNTLG],
                          [DMAG], GMMCT, DMVI );
                ENDIF;
                [DGMVC(BC,SUB)] := -TRANS([PHIG(BC)]) * [DMAG];
                [DGMCB(BC,SUB)] := (1) [DGMVC(BC,SUB)];
              ENDIF;
              CALL MIST ( NITER, BC, SUB, GSIZEB, [QHHLFL(BC,SUB)],
                      [QHC(BC,SUB)], [QHG(BC,SUB)],
                      [MII], [MCC(BC,SUB)], [MHC(BC,SUB)],
                      LAMBDA, [PHIG(BC)], [DTMSB(BC,SUB)],
                      [EMSB(BC,SUB)], [AMSB(BC,SUB)],
                      [RMS(BC,SUB)],  [NLCR(BC,SUB)] );
$               CALL MMIST ( NITER, BC, SUB, GSIZEB, [QHHLFL(BC,SUB)],$
$                       [QHC(BC,SUB)], [QHG(BC,SUB)], [MII],$
$                       [MCC(BC,SUB)], [MHC(BC,SUB)], LAMBDA,$
$                      [PHIG(BC)], [DTMSB(BC,SUB)],[EMSB(BC,SUB)],$
$                       [AMSB(BC,SUB)], [RMS(BC,SUB)] );$
              [DTMS(BC,SUB)] := (1) [DTMSB(BC,SUB)];
              [EMS(BC,SUB)]  := (1) [EMSB(BC,SUB)];
              [AMS(BC,SUB)]  := (1) [AMSB(BC,SUB)];
            ENDIF;
            CALL ASEFLUT ( NITER, BC, SUB, ESIZE(BC), GSIZEB, NRSET,
```

```
                    [MHHFL(BC,SUB)],
                    [KRHHFL(BC,SUB)], [BHHFL(BC,SUB)],
                    [EMS(BC,SUB)], [DTMS(BC,SUB)],
                    [AMS(BC,SUB)], [RMS(BC,SUB)],
                    [MHC(BC,SUB)], [PHIG(BC)],
                    [NLCR(BC,SUB)], [PHFLO(BC,SUB)],
                    [PLROG(BC,SUB)],
                    ASELAMBD, ASEMARG, CONST );
        ENDIF;
INSERT 976
    IF NOT BCFASE THEN
INSERT 977
    ELSE
      IF NUMOPTBC > 1 CALL NULLMAT ([UF], [AF], [UTRANF], [UFREQF]);
    ENDIF;
INSERT 1019
      IF NOT BCFASE THEN
INSERT 1024
      ENDIF;
INSERT 1072
        IF NOT BCFASE THEN
INSERT 1073
        ENDIF;
INSERT 1106
        IF NOT BCFASE THEN
INSERT 1107
        ENDIF;
INSERT 1112
    IF NOT BCFASE THEN
INSERT 1113
    ELSE
      IF NUMOPTBC > 1 CALL NULLMAT ( [UN], [AN] );
    ENDIF;
INSERT 1136
      IF NOT BCFASE THEN
INSERT 1145
      ENDIF;
INSERT 1175
      IF NOT BCFASE THEN
INSERT 1176
      ENDIF;
INSERT 1180
    IF NOT BCFASE THEN
INSERT 1182
    ELSE
```

```
      IF NUMOPTBC > 1 CALL NULLMAT([UG(BC)],[AG(BC)],[UAG(BC)],[AAG(BC)]);
    ENDIF;
INSERT 1212
      IF NOT BCFASE THEN
INSERT 1216
      ENDIF;
INSERT 1242
      IF NOT BCFASE THEN
INSERT 1243
      ENDIF;
INSERT 1466
            CALL ASEDRV ( BC, SUB, LASE );
            LASE := TRUE;
INSERT 1467
             IF NOT LASE THEN
INSERT 1474
             ELSE
               CALL FLUTDRV ( BC, SUB, LOOP );
               IF ASEOPTDB THEN
                 PRINT("LOG=('        GM SENSITIVITY')");
                 CALL MAKDVU ( NITER, NDV, GLBDES, [GTMP],
                        [DKUG], GMKCT, DKVI );
                 [DGKVB(BC)] := -TRANS([GTMP]) * [DKUG];
                 CALL MAKDVU ( NITER, NDV, GLBDES, [GTMP],
                        [DMAG], GMMCT, DMVI );
                 [DGMVB(BC)] := -TRANS([GTMP]) * [DMAG];
                 [DGMV(BC)] := (1)[DGMVB(BC)];
                 [DGKV(BC)] := (1)[DGKVB(BC)];
               ENDIF;
               CALL ASESENS ( NITER, BC, SUB, ESIZE(BC), GSIZEB,
                      NDV,
                      [MHHFL(BC,SUB)], [KRHHFL(BC,SUB)],
                      [BHHFL(BC,SUB)],
                      [DGKV(BC)], [DGMV(BC)],
                      [DGMVC(BC,SUB)],
                      [EMS(BC,SUB)], [DTMS(BC,SUB)],
                      [AMS(BC,SUB)], [RMS(BC,SUB)],
                      [MHC(BC,SUB)], [PHIG(BC)],
                      [PHFLO(BC,SUB)], [PLROG(BC,SUB)],
                      ASELAMBD, ASEMARG, CONST, [AMAT] );
             ENDIF;
INSERT 1475
             IF NOT LASE THEN
INSERT 1482
             ELSE
```

```
                CALL FLUTDRV ( BC, SUB, LOOP );
                IF ASEOPTDB THEN
                  PRINT("LOG=('        GM SENSITIVITY')");
                  CALL MAKDVU ( NITER, NDV, GLBDES, [PHIG(BC)],
                           [DKUG], GMKCT, DKVI );
                  [DGKVB(BC)] := -TRANS([PHIG(BC)]) * [DKUG];
                  CALL MAKDVU ( NITER, NDV, GLBDES, [PHIG(BC)],
                           [DMAG], GMMCT, DMVI );
                  [DGMVB(BC)] := -TRANS([PHIG(BC)]) * [DMAG];
                  [DGMV(BC)] := (1)[DGMVB(BC)];
                  [DGKV(BC)] := (1)[DGKVB(BC)];
                ENDIF;
                CALL ASESENS ( NITER, BC, SUB, ESIZE(BC), GSIZEB,
                         NDV,
                         [MHHFL(BC,SUB)], [KRHHFL(BC,SUB)],
                         [BHHFL(BC,SUB)],
                         [DGKV(BC)], [DGMV(BC)],
                         [DGMVC(BC,SUB)],
                         [EMS(BC,SUB)], [DTMS(BC,SUB)],
                         [AMS(BC,SUB)], [RMS(BC,SUB)],
                         [MHC(BC,SUB)], [PHIG(BC)],
                         [PHFLO(BC,SUB)], [PLROG(BC,SUB)],
                         ASELAMBD, ASEMARG, CONST, [AMAT] );
               ENDIF;
INSERT 1936
          IF NITER+1 = NITERUP+NUP THEN
            ASEOPTDB := TRUE;
            NITERUP := NITERUP + NUP;
          ELSE
            ASEOPTDB := FALSE;
          ENDIF;
$                                                         $
INSERT 1972
$                                                         $
$  FIND VINIG CARDS IN THE BULK DATA AND UPDATE DESIGN VARIABLES
$
$                                                         $
  DVFLG := FALSE;
  IF BCFASE AND NUMOPTBC = 0 THEN
    CALL UPDV ( VINIG, GLBDES, NDV, DVFLG, [DELDV] );
    CALL DFREL ( ASELAMBD, 2 );
    CALL DFREL ( LAMBDA, 2 );
  ENDIF;
INSERT 1977
$                                                         $
```

```
$       MAKE ASEBC FLAG FOR THE CURRENT BOUNDARY CONDITION                   $
$                                          $
      CALL ASEBC ( BC, BCFASE);
INSERT 2020
IF NOT DVFLG THEN
INSERT 2144
ENDIF;
INSERT 2148
IF NOT DVFLG THEN
INSERT 2198
ENDIF;
INSERT 2202
IF NOT DVFLG THEN
INSERT 2217
ENDIF;
INSERT 2232
IF NOT DVFLG THEN
INSERT 2238
ENDIF;
INSERT 2247
IF NOT DVFLG THEN
INSERT 2253
ENDIF;
INSERT 2254
      IF BCFASE THEN
        IF NOT DVFLG THEN
$                                          $
$          CREATE GENERALIZED MATRICES OF THE BASELINE              $
$                                          $
          CALL GMAT ( , BC, LAMBDA, [MII], [GMB(BC)], [GKB(BC)] );
          [PHIAB(BC)] := (1) [PHIA];
          CALL AGRECOV ( [PHIA], [PHIO], [PHIF], [PHIN], [UM], [TMP1]);
          [PHIG(BC)] := (1) [TMP1];
        ELSE
$                                          $
$          CREATE GENERALIZED MATRICES OF THE MODIFIED STRUCTURE          $
$                                          $
          CALL GMGKUPD ( ,  , NDV, DVFLG, [DGMV(BC)], [DGKV(BC)],
                    [DELDV], GLBDES, [GMB(BC)], [GKB(BC)],
                    [DELGM], [DELGK], [GM(BC)], [GK(BC)],
                    [DGMVC(BC,SUB)], [MICB(BC,SUB)],
                    [MIC(BC,SUB)] );
$                                          $
$          REDUCED MODAL ANALYSIS OF THE MODIFIED STRUCTURE          $
$                                          $
```

```
        CALL RMODE (  , BC, HSIZE(BC), USET(BC), [GK(BC)],
                  [GM(BC)], , ,LAMBDA, [PSI], [MII], IPA );
        CALL OFPMROOT ( , BC, NUMOPTBC, LAMBDA );
        [PHIA] := [PHIAB(BC)] * [PSI];
        CALL AGRECOV ( [PHIA], [PHIO], [PHIF], [PHIN], [UM], [TMP1]);
        [PHIG(BC)] := (1) [TMP1];
      ENDIF;
    ENDIF;
INSERT 2440
$                                                    $
$          ASEDRV MAKE FLAG: LASE = TRUE, IF METHOD = ASE          $
$                                                    $
        CALL ASEDRV ( BC, SUB, LASE );
        IF NOT LASE OR NOT DVFLG THEN
INSERT 2448
        ELSE
$                                                    $
$            UPDATE MATRICES FOR FLUTTER                    $
$                                                    $
          CALL MGFL (  , BC, SUB, HSIZE(BC), ESIZE(BC),
                  LAMBDA, [MII], [PSI], [GM2(BC,SUB)],
                  [GK2(BC,SUB)], [GB2(BC,SUB)],
                  [MHHFL(BC,SUB)],  [KRHHFL(BC,SUB)],
                  [BHHFL(BC,SUB)], [DTMSB(BC,SUB)],
                  [EMSB(BC,SUB)], [AMSB(BC,SUB)],
                  [DTMS(BC,SUB)], [EMS(BC,SUB)],
                  [AMS(BC,SUB)],
                  [MIC(BC,SUB)], [MHC(BC,SUB)] );
        ENDIF;
        IF NOT LASE THEN
INSERT 2451
        ELSE
          IF NOT DVFLG THEN
$                                                    $
$              MAKE CONTROL AND GUST MODES                    $
$                                                    $
            CALL CNTMOD (  , BC, SUB, [AJC], [SKJ], [SCNTLK], [ACNTLK],
                   [SCNTLG], [ACNTLG], REUNMK, [PHIKH],
                   [PHIG(BC)], [MGG], [QHC(BC,SUB)],
                   [MCC(BC,SUB)], [MICB(BC,SUB)],
                   CONTRFL, ISYM );
            [MIC(BC,SUB)] := (1) [MICB(BC,SUB)];
            [MHC(BC,SUB)] := (1) [MICB(BC,SUB)];
            CALL GSTMOD (  , BC, SUB, [QGK], REUNMK, [PHIKH],
                     [QHG(BC,SUB)] );
```

```
$                                                        $
$            MAKE SENSITIVITIES FOR CONTROL MODES MASS COUPLING        $
$                                                        $
           IF CONTRFL THEN
             IF ISYM = 1 THEN
               CALL MAKDVU ( 1, NDV, GLBDES, [SCNTLG],
                         [DMAG], GMMCT, DMVI );
             ELSE
               CALL MAKDVU ( 1, NDV, GLBDES, [ACNTLG],
                         [DMAG], GMMCT, DMVI );
             ENDIF;
             [DGMVC(BC,SUB)] := -TRANS([PHIG(BC)]) * [DMAG];
             [DGMCB(BC,SUB)] := (1) [DGMVC(BC,SUB)];
           ENDIF;
           CALL MIST ( 0, BC, SUB, GSIZEB, [QHHLFL(BC,SUB)],
                   [QHC(BC,SUB)], [QHG(BC,SUB)],
                   [MII], [MCC(BC,SUB)], [MHC(BC,SUB)],
                   LAMBDA, [PHIG(BC)], [DTMSB(BC,SUB)],
                   [EMSB(BC,SUB)], [AMSB(BC,SUB)],
                   [RMS(BC,SUB)],  [NLCR(BC,SUB)] );
$            CALL MMIST ( 0, BC, SUB, GSIZEB, [QHHLFL(BC,SUB)],$
$                   [QHC(BC,SUB)], [QHG(BC,SUB)], [MII],$
$                   [MCC(BC,SUB)], [MHC(BC,SUB)], LAMBDA,$
$                   [PHIG(BC)], [DTMSB(BC,SUB)],[EMSB(BC,SUB)],$
$                   [AMSB(BC,SUB)], [RMS(BC,SUB)] );$
           [DTMS(BC,SUB)] := (1) [DTMSB(BC,SUB)];
           [EMS(BC,SUB)]  := (1) [EMSB(BC,SUB)];
           [AMS(BC,SUB)]  := (1) [AMSB(BC,SUB)];
           CALL BDMAT ( , BC, SUB, HSIZE(BC), ESIZE(BC),
                   [GMB(BC)], [GKB(BC)],
                   [MHHFL(BC,SUB)], [KHHFL(BC,SUB)],
                   [BHHFL(BC,SUB)], [KRHHFL(BC,SUB)],
                   [GM2(BC,SUB)],[GK2(BC,SUB)],
                   [GB2(BC,SUB)] );
         ENDIF;
         CALL ASEFLUT ( 0, BC, SUB, ESIZE(BC), GSIZEB, NRSET,
                 [MHHFL(BC,SUB)],
                 [KRHHFL(BC,SUB)], [BHHFL(BC,SUB)],
                 [EMS(BC,SUB)], [DTMS(BC,SUB)],
                 [AMS(BC,SUB)], [RMS(BC,SUB)], [MHC(BC,SUB)],
                 [PHIG(BC)], [NLCR(BC,SUB)],
                 [PHFLO(BC,SUB)], [PLROG(BC,SUB)],
                 ASELAMBD, ASEMARG );
       ENDIF;
INSERT 2514
```

```
  IF NOT BCFASE THEN
INSERT 2515
  ELSE
    IF NUMOPTBC > 1 CALL NULLMAT ( [UF], [AF] );
  ENDIF;
INSERT 2549
    IF NOT BCFASE THEN
INSERT 2554
    ENDIF;
INSERT 2594
      IF NOT BCFASE THEN
INSERT 2598
      ENDIF;
INSERT 2624
      IF NOT BCFASE THEN
INSERT 2625
      ENDIF;
INSERT 2630
  IF NOT BCFASE THEN
INSERT 2631
  ELSE
    IF NUMOPTBC > 1 CALL NULLMAT ( [UN], [AN] );
  ENDIF;
INSERT 2648
    IF NOT BCFASE THEN
INSERT 2657
    ENDIF;
INSERT 2685
    IF NOT BCFASE THEN
INSERT 2686
    ENDIF;
INSERT 2690
  IF NOT BCFASE THEN
INSERT 2692
  ELSE
    IF NUMOPTBC > 1 CALL NULLMAT([UG(BC)],[AG(BC)],[UAG(BC)],[AAG(BC)]);
  ENDIF;
INSERT 2713
    IF NOT BCFASE THEN
INSERT 2717
    ENDIF;
INSERT 2737
    IF NOT BCFASE THEN
INSERT 2738
    ENDIF;
```

```
INSERT 2741
$                                                    $
$   CREATE SENSITIVITIES OF GENERALIZED MATRICES                      $
$                                                    $
    IF BCFASE AND ASEDBMAK THEN
      PRINT("LOG=('        GM SENSITIVITY')");
      CALL MAKDVU ( 1, NDV, GLBDES, [PHIG(BC)], [DKUG], GMKCT, DKVI );
      [DGKVB(BC)] := -TRANS([PHIG(BC)]) * [DKUG];
      CALL MAKDVU ( 1, NDV, GLBDES, [PHIG(BC)], [DMAG], GMMCT, DMVI );
      [DGMVB(BC)] := -TRANS([PHIG(BC)]) * [DMAG];
      [DGMV(BC)] := (1)[DGMVB(BC)];
      [DGKV(BC)] := (1)[DGKVB(BC)];
      CALL DFREL ( ASEUP, 2 );
    ENDIF;
```

Chapter 3

ASE Solution Control

Since we are not able to change the solution control packet in this project, the ASE solution options are defined by a combination of the regular FLUTTER solution control command, a modified version of the FLUTTER template, and a new ASESOL template.

All the parameters of the regular FLUTTER solution control command, except CONTROL and TFL are applicable in the ASE discipline. The ID number defined by the FLCOND parameter refers to a FLUTTER template. The FLUTTER template definition is expanded to allow the method in the 3rd field to be "ASE". When METHOD $\neq$ ASE, the regular flutter discipline is use. When METHOD = ASE, the ASEDRV module looks for an ASESOL template with the ID number defined by FLCOND. The ASESOL template defines all options of the ASE discipline that actually belong to the solution control packet. These are the selection of run option, type of ASE analysis, the ID numbers of the bulk data entries which define the main parameters of the rational aerodynamic approximation (RAA), control model, and gust conditions.

Chapter 4

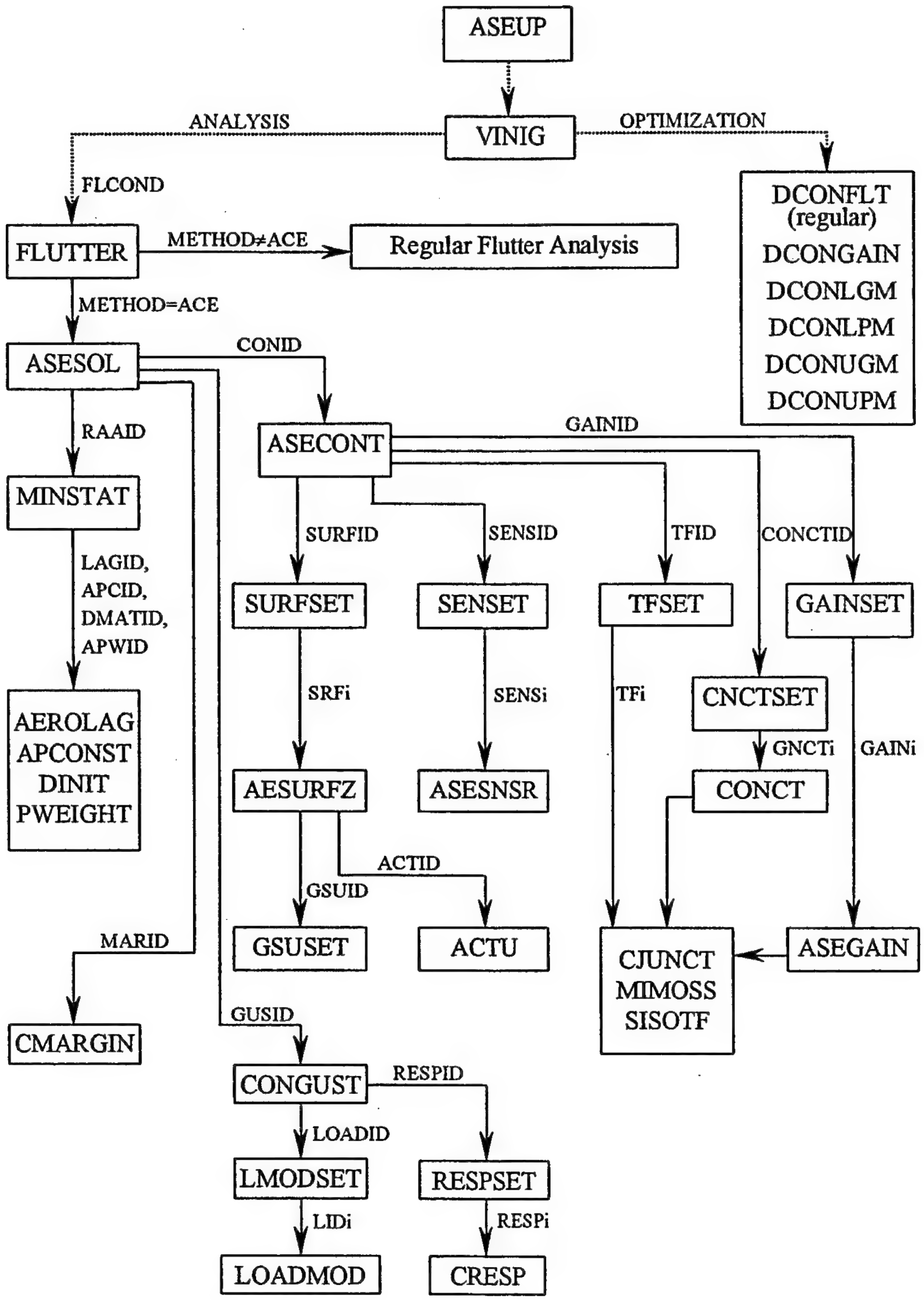
The Bulk Data Packet

4.1 Summary of new bulk data entries

The bulk data entries already added for the new ASE module are:

ACTU	definition of an actuator transfer function
AEROLAG	definition of approximation lag values (roots)
AEROGND	non-default Roger's approximation roots
AESURFZ	definition of an aerodynamic control surface
APCONST	definition of default approximation constraints
APCNSND	definition of non-default approximation constraints
ASECONT	basic parameters of the control system
ASEGAIN	definition of control gains
ASESNSR	definition of sensors
ASESOL	ASE solution control parameters
ASEUP	ASE modal data base options
CJUNCT	definition of a MIMO junction control element
CNCTSET	set of fixed control element connections
CONCT	fixed connections of control elements
CONGUST	continuous gust parameters
CMARGIN	parameters for control margin analysis
CRESP	definition of sensors at which gust response is requested
DCONGAIN	definition of an open-loop aeroelastic gain constraint
DCONLGM	definition of an lower gain margin constraint
DCONLPM	definition of an lower phase margin constraint
DCONUGM	definition of an upper gain margin constraint
DCONUPM	definition of an upper phase margin constraint
DINIT	initial $[D]$ for minimum-state approximation
GAINSET	definition of a set of gains
MIMOSS	definition of a MIMO controller
MINSTAT	parameters for aerodynamic approximation
PWEIGHT	aero approximation weighting parameters
RESPSET	definition of a set of gust response points
SENSET	definition of a set of sensors
SISOTF	definition of a SISO controller
SURFSET	definition of a set of control surfaces
TFSET	selection of a set of transfer functions
VINIG	change initial values of design variables

The hierarchy of the new bulk data entries and the parameters defining the associated ID numbers are shown in Figure 4.1. A detailed description of the data entries is given in Section 4.2.



4.2 Bulk data entries

Input Data Entry: **ACTU** Actuator transfer function

Description: Defines the actuator transfer function.

Format and Example:

1	2	3	4	5	6	7	8	9	10
ACTU	ID	A0	A1	A2					

ACTU	10	0.8	0.5	0.2					

Field	Contents
ID	Identification number (Integer > 0)
Ai	The denominator coefficients in the actuator transfer function (Real)

$$\frac{\delta}{u_{ac}} = \frac{A_0}{s^3 + A_2 * s^2 + A_1 * s + A_0}$$

Remarks:

1. All ID numbers of ACTU, ASESNSR, CJUNCT, CRESP, MIMOSS, SISOTF entries must be distinct.

2. The actuators are selected in the AESURF entry.

3. The order of the denominator polynomial is larger than the numerator order by at least 3 to avoid noise going through.

4. Higher-order actuator can be defined by adding a transfer function in series (using the SISOTF entry).

. Input Data Entry: **AEROLAG** Aerodynamic approximation roots

Description: Defines the approximation roots for rational approximation of unsteady aerodynamic matrices.

Format and Example:

1	2	3	4	5	6	7	8	9	10
AEROLAG	SID	NLAG	R1	R2	R3	R4	R5	R6	CONT
CONT	R7	R8	-etc-						

AEROLAG	10	4	-0.2	-0.5	-1.0	-2.0			

Field	Contents
SID	Set identification number (Integer > 0)
NLAG	Number of approximation lag terms (Integer ≥ 0)
Ri	NLAG Distinct root values (Real < 0, or blanks)

Remarks:

1. SID is selected by the MINSTAT data entry.

2. With minimum state approximation, the number of roots is n_L = NLAG. With Roger's approximation $n_L = \sum_{i=1}^{n_h} \mathrm{NLAG}_i$ where n_h is the number of structural modes. NLAG_i=NLAG, unless defined otherwise in an AEROGND entry with the same SID.

3. If R1 is blank, the approximation root values are calculated by

$$R_i = -1.7k_{max}\left(\frac{i}{\mathrm{NLAG}+1}\right)^2$$

where k_{max} is the maximal reduced frequency of the aerodynamic data.

Input Data Entry: **AEROGND** Non-default Roger's approximation roots (optional).

Description: Defines Roger's aerodynamic roots which replace those defined in the AEROLAG entry.

Format and Example:

1	2	3	4	5	6	7	8	9	10
AEROGND	SID	NDRI	NLAGI	R1	R2	etc.			

AEROGND	10	2	-0.3	-0.7					

Field	Contents
SID	Set identification number (Integer > 0).
NDRI	Index of a row in the unsteady aerodynamic matrix for which the default Roger's approximation roots (defined in the AEROLAG entry) are replaced by those defined here (Integer > 0).
NLAGI	Now number of approximation lag terms (0 < Integer ≤ 5).
Ri	NLAGi distinct root values (Real < 0).

Remarks:

1. SID must be the same as that of the AEROLAG entry which defines the default roots.
2. All NDRI indices in AEROGND with the same SID must be distinct.

Input Data Entry: **AESURFZ** Aerodynamic control surface

Description: Specifies an aerodynamic control surface.

Format and Example:

1	2	3	4	5	6	7	8	9	10
AESURF	LABEL	TYPE	ACOD	CID	FBOXID	LBOXID	GSUID	ACTID	
AESURF	SURF1	SYM	30	40	50	60	70	80	

Field	Contents
LABEL	Unique alphanumeric string of up to eight characters used to identify the control surface
TYPE	Surface type (Character)(Remark 2) SYM symmetric surface ANTISYM anti-symmetric surface ASYM asymmetric surface
ACID	Identification number of the aircraft component (CAERO6) on which the surface lies (Integer > 0)
CID	Identification number of a rectangular coordinate system whose y- axis defines the hinge line of the control surface (Integer > 0, or blank)
FBOXID	First aero box on the control surface relative to ACID (Integer > 0)
LBOXID	Last aero box on the control surface relative to ACID (Integer > 0)
GSUID	Identification number of the GSUSET entry which defines the surface structure grid points (Integer > 0)
ACTID	Identification number of the ACTU entry defining the transfer function of the actuator attached to this control surface (Integer > 0)

Remarks:

1. The LABEL is arbitrary, but all labels must be unique.

2. The asymmetric surface, TYPE = ASYM is not currently available. Pitch controllers are TYPE = SYM while yaw and roll controllers are TYPE = ANTISYM.

3. The aerodynamic box numbering scheme is illustrated on the CAERO1 Bulk Data entry.

4. This is an old template with 2 new fields, GSUID and ACTID.

Input Data Entry: **APCONST** Definition of approximation constraints

Description: Divides the aerodynamic matrix into blocks and defines default constraints for each block.

Format and Example:

1	2	3	4	5	6	7	8	9	10
APCONST	SID	DA0	DA1	DA2	NRP	NCP	FR(1)	-etc-	CONT
CONT	FR(NRP)	FC(1)	-etc-	FC(NCP)					

1	2	3	4	5	6	7	8	9	10
APCONST	21	1	-1	-1	1	3	1	1	+AB
+AB	24	26							+AC

Field	Contents
SID	Set identification number (Integer > 0)
NRP	Number of row partitions (Integer > 0)
FR(i)	Index of first row of the i-th row partition (Integer > 0)
NCP	Number of column partions (Integer > 0)
FC(j)	Index of first column of the j-th column partition (Integer > 0)
DA0	Defines default steady fit constraint (Integer ≥ 0) = 0: non constraint ; = 1: match data at $k_1 = 0$
DA1	Defines default constraint on the imaginary part (Integer) <0: no constraint; = 0: set A1 = 0; = 1: match imaginary part of data on k_{max}; > 1: match imaginary part of data at k_{DA1}
DA2	defines default constraint on real part (Integer); < 0: no constraint; = 0: set A2 = 0; = 1: match real part of data at k_{max}; > 1: match real part of data at k_{DA2}

Remarks:

1. SID is reflected by the MINSTAT data entry.

2. DA0 = 1 is recommended for all the aerodynamic terms.

3. DA1 < 0 and DA2 < 0 usually yield best results.

4. DA1 = 0 might cause large modeling errors.

5. The code assumes DA2 = 0 for all the gust columns.

Input Data Entry: **APCNSND** Definition of approximation constraints

Description: Defines one set of non-default constraints for each block.

Format and Example:

1	2	3	4	5	6	7	8	9	10
APCONST	SID	IP	JP	PA0	PA1	PA2			CONT
CONT	IP	JP	PA0	PA1	PA2				CONT
CONT	IP	JP	PA0	-etc-					

1	2	3	4	5	6	7	8	9	10
APCONST	21	1	3	1	1	0			

Field	Contents
SID	Set identification number (Integer > 0)
IP,JP	Indices of a block which assumes non-default constraints (0 < Integers < NRP, NCP)
PA0,PA1,PA2	replace DA0, DA1, DA2

Remarks:

1. SID is reflected by the MINSTAT data entry.

Input Data Entry: **ASECONT** Basic parameters of the control system

Description: Defines the basic parameters of the control system.

Format and Example:

1	2	3	4	5	6	7	8	9	10
ASECONT	SID	SURFID	SENSID	TFID	GAINID	CONCTID			

ASECONT	10	20	30	40	50	60			

Field	Contents
SID	Set identification number (Integer > 0)
SURFID	Identification number of the SURFSET entry specifying the control surfaces (Integer > 0)
SENSID	Identification number of the SENSET entry specifying the sensors (Integer > 0)
TFID	identification number of the TFSET entry specifying the control transfer functions (Integer > 0, or blank)
GAINID	Identification number of the GAINSET entry specifying the connection through gains (Integer > 0, or blank)
CONCTID	Identification number of the CONCT entry that defines control connection without gains (Integer > 0, or blank)

Remarks:

1. SID is selected by the CONID parameter of the ASESOL data entry.
2. If TFID is blank, no transfer functions are defined between sensor's outputs and actuator's inputs.
3. The actuators are specified in the AESURF entries.
4. If GAINID is blank, no gains are defined.
5. If CONCTID is blank, no connections are defined.

Input Data Entry: **ASEGAIN** ASE control gains

Description: Defines control gains of the ASE control system.

Format and Example:

1	2	3	4	5	6	7	8	9	10
ASEGAIN	ID	OTFID	CO	ITFID	CI	GAIN			

ASEGAIN	10	1	1	2	2	0.6			

Field	Contents
ID	Identification number (Integer > 0)
OTFID	Identification number of the downstream control element defined in CJUNCT, MIMOSS, SISOTF or ASESNSR entry (Integer > 0)
CO	The output component of OTFID (Integer > 0)
ITFID	Identification number of the upstream control element defined in CJUNCT, MIMOSS, SISOTF or ACTU entry (Integer > 0)
CI	The input component of ITFID (Integer > 0)
GAIN	The connection gain (Real)

Remarks:

1. All ID numbers of ASEGAIN entries must be unique.
2. ID is selected by the GAINSET entry.
3. GAIN defines a term in $\{u\}=[G]\{y\}$ which connects the CO-th output of OTFID to the CI-th input of ITFID.

Input Data Entry: **ASESNSR** Sensor

Description: Defines a sensor.

Format and Example:

1	2	3	4	5	6	7	8	9	10
ASESNSR	ID	TYPE	SGID	SC					

ASESNSR	10	2	7	6					

Field	Contents
ID	Identification number (Integer > 0)
TYPE	Sensor type (Integer) 0 - displacement 1 - rate 2 - acceleration
SGID	Identification number of the grid or scalar point at which the sensor is mounted (Integer > 0)
SC	Component number (1 - 6, or blank)

Remarks:

1. All ID numbers of ACTU, ASESNSR, CJUNCT, CRESP, MIMOSS, SISOTF entries must be distinct.

2. Sensors are selected in the SENSET entry.

Input Data Entry: **ASESOL** Aeroservoelastic solution control

Description: Defines the ASE run option and refers to the bulk data entries that define the specific analysis parameters.

Format and Example:

1	2	3	4	5	6	7	8	9	10
ASESOL	SID	RAAID	CONID	FLASE	MARID	GUSID			

ASESOL	5	20	30	1	40	50			

Field	Contents
SID	Set identification number (Integer > 0)
RAAID	Identification number of a MINSTAT set specifying parameters for rational aerodynamic approximation (Integer > 0)
CONID	Identification number of a ASECONT set specifying parameters of the control system (Integer $\geq$ 0, or blank)
FLASE	If >0, perform flutter analysis. Find ASE roots at conditions defined in the FLUTTER entry with the same SID
MARID	Identification number of a CMARGIN set specifying parameters for control margin analysis (Integer $\geq$ 0, or blank)
GUSID	Identification number of a CONGUST set specifying parameters of the continuous gust cases. (Integer $\geq$ 0, or blank)

Remarks:

1. SID is selected by the FLCOND command of flutter solution control.
2. If CONID is blank or zero, no control system is used.
3. If MARID is blank or zero, no control margins are analyzed.
4. If GUSID is blank or zero, no gust columns are included in the rational approximation.

Input Data Entry: **ASEUP** ASE modal data base options

Description: Defines ASE data-base creation/usage options.

Format and Example:

1	2	3	4	5	6	7	8	9	10
ASEUP	ASEDB	NUP							

ASEUP	1								

Field	Contents
ASEDB	Flag for creation or use of a modal data base (Integer): 0 or blank, no modal data base is created or used 1, create modal data base for subsequent ASE runs (applicable only in FINAL ANALYSIS) 2, use modal data base created in a previous run
NUP	Frequency of modal data base updates during design optimization: 0 or blank, no data-base updates are performed $i > 0$, update the discrete model every i iterations and reconstruct the modal data base

Remarks:

1. If ASEDB = 2, the run should be set up as a restart run with an OLD data base.

Input Data Entry: **CJUNCT** Junction control element

Description: Defines the MIMO junction control element.

Format and Example:

1	2	3	4	5	6	7	8	9	10
CJUNCT	ID	NU	NY	D11	D12	-etc-	D(1,NY)	D21	CONT
CONT	D22	-etc-	D(NU,NY)						

CJUNCT	90	3	1	1.0	-1.0	0.5			

Field	Contents
ID	Identification number (Integer > 0)
NU,NY	Number of inputs and outputs in $\{y\} = [D]\{u\}$, (Integer > 0)
Dij	Element of $[D]$, (Real)

Remarks:

1. All ID numbers of ACTU, ASESNSR, CJUNCT, CRESP, MIMOSS, SISOTF entries must be distinct.

Input Data Entry: **CMARGIN** Control margin analysis parameters

Description: Defines the ranges of interest for finding gain margins, phase margins, and singular values.

Format and Example:

1	2	3	4	5	6	7	8	9	10
CMARGIN	SID	UGM	LGM	UPM	LPM	USVI	USVO		

CMARGIN	40	8.0	-8.0	60.	-60.	1.5	0.		

Field	Contents
SID	Set identification number (Integer > 0)
UGM	Upper limit (in db) for search of positive gain margins (Real ≥ 0, or blank)
LGM	Lower limit (in db) for search of positive gain margins (Real ≤ 0, or blank)
UPM	Upper limit (in degrees) for search of positive phase margins (Real ≥ 0, or blank)
LPM	Lower limit (in degrees) for search of positive phase margins (Real ≤ 0, or blank)
USVI	Upper limit for search of input MIMO margins in terms of smallest singular values (Real ≥ 0, or blank)
USVO	Upper limit for search of output MIMO margins in terms of smallest singular values (Real ≥ 0, or blank)

Remarks:

1. SID is selected by the MARID parameter of the ASESOL data entry.

2. If any of the parameters in the template is blank or zero, the associated control margin is not calculated.

3. When a parameter is not equal to zero, all control margins within specified limits will be calculated.

Input Data Entry: **CNCTSET** Set of fixed control element connections

Description: Defines the set of connections of transfer functions of an ASE case.

Format and Example:

1	2	3	4	5	6	7	8	9	10
CNCTSET	SID	CNCT1	CNCT2	CNCT3	CNCT4	CNCT5	CNCT6	CNCT7	CONT
CONT	CNCT8	-etc-							

GAINSET	10	4	2	5	1	3			

Field	Contents
SID	Set identification number (Integer > 0)
CNCTi	Identification numbers of an CONCT entry defining the connections of transfer functions of the control system

Remarks:

1. SID is selected in the ASECONT data entry.

Input Data Entry: **CONCT** Fixed connections of a control element

Description: Defines fixed connections of transfer functions of the ASE control system.

Format and Example:

1	2	3	4	5	6	7	8	9	10
CONCT	ID	OTFID	CO	ITFID	CI				

CONCT	10	1	1	2	2				

Field	Contents
ID	Identification number (Integer > 0)
OTFID	Identification number of the downstream control element defined in CJUNCT, MIMOSS, SISOTF or ASESENS entry (Integer > 0)
CO	The output component of OTFID (Integer > 0)
ITFID	Identification number of the upstream control element defined in CJUNCT, MIMOSS, SISOTF or ACTU entry (Integer > 0)
CI	The input component of ITFID (Integer > 0)

Remarks:

1. All ID numbers of CONCT entries must be unique.
2. ID is selected by the CNCTSET entry.
3. The inputs of all downstream control elements and the outputs of all upstream control elements in this entry are eliminated from the input and output vectors and cannot be used for defining gains in the ASEGAIN entry.

Input Data Entry: **CONGUST** Gust response parameters

Description: Defines parameters for gust response analysis

Format and Example:

1	2	3	4	5	6	7	8	9	10
CONGUST	SID	LG	GURMS	GUFILT	LPASS	LOADID	RESPID		
CONGUST	50	2500	25.0	DRY	100.	10	20		

Field	Contents
SID	Set identification number (Integer > 0)
LG	Scale of turbulence (Real > 0), see Remark 2
GURMS	RMS value of the gust velocity (Real > 0)
GUFILT	Definition of gust filter (Character) DRY - exact representation of Dryden's gust spectrum VK1, VK2 - approximate representation of Von Karman's gust spectrum, see Remark 3
LPASS	Parameter a in the low-pass filter a/(s+a) that multiplies the gust filter (Real > 0), see Remark 4
LOADID	Identification number of the LMODSET entry specifying the load modes (Integer ≥ 0, or blank), see Remark 5
RESPID	Identification number of the RESPSET entry specifying sensors at which response is requested (Integer ≥ 0, or blank) , see Remark 6

Remarks:

1. SID is selected by the GUSID parameter in the ASESOL data entry.

2. Typical atmospheric scale of turbulence is Lg = 2500 ft.

3. Gust filters are given in the Theoretical Manual. VK1 is adequate for $\omega \leq 20$ V / Lg. VK2 is adequate for $\omega \leq 200$ V / Lg.

4. LPASS should be larger than the maximal frequency of interest (in rad/sec).

5. If LOADID = 0 or blank, no section loads are requested.

6. If RESPID = 0 or blank, no discrete responses are requested. The sensors selected by the SENSET entry referred to by RESPID are not necessarily actual sensors.

Input Data Entry: **CRESP** Sensor

Description: Defines a sensor at which gust response is requested.

Format and Example:

1	2	3	4	5	6	7	8	9	10
CRESP	ID	TYPE	SGID	SC					

CRESP	10	2	7	6					

Field	Contents
ID	Identification number (Integer > 0)
TYPE	Sensor type (Integer) 0 - displacement 1 - rate 2 - acceleration
SGID	Identification number of the grid or scalar point at which the sensor is mounted (Integer > 0)
SC	Component number (1 - 6, or blank)

Remarks:

1. All ID numbers of CRESP entries must be distinct.
2. Sensors are selected in the RESPSET entry.

Input Data Entry: **DCONGAIN**

Description: Defines an open-loop aeroelastic gain constraint in the form of a table:

$$\frac{1}{GFACT}\left(\sum_{i=1}^{n_{cs}} G_{p,i}G_i + G_{req}\right) \le 0$$

Format and Example:

1	2	3	4	5	6	7	8	9	10
DCONGAIN	SID	VTYPE	GFAC	GAINID	V1	GPV1	V2	GPV2	CONT
CONT	V3	GPV3	V4	GPV4	-etc-				

1	2	3	4	5	6	7	8	9	10
DCONGAIN	1		1.0	902	0.0	4.0	12060.	4.0	CONT
CONT	15000.	1.0							

Field	Contents
SID	Constraint set identification, the constraint are referenced by the design constraint ID in Solution Control (Integer > 0)
VTYPE	Nature of the velocity referred in the table. Either TRUE for true velocity or EQUIV for equivalent air speed. Default = TRUE
GFACT	Constraint scaling factor (Real > 0.0, Default=1.0)
GAINID	Identification number of the GAINSET entry specifying the considered actuator gains G_i (Integer > 0)
VI	Velocity value (Real ≥ 0.0)
GPVI	Required vehicle gain value for zero frequency (Real)

Remarks:

1. Open-loop vehicle gain constraints are selected in Solution Control with the discipline option: DCON=SID.

2. The VI must be in either ascending or descending order.

3. At least two pairs must be entered.

4. Only gains associated with a single roll sensor in anti-symmetric maneuver are currently considered. The sensor ID must be the smallest one in the associated SENSET entry.

5. The control system must include gain elements at the inputs of all actuators considered.

Input Data Entry: **DCONLGM**

Description: Defines an lower gain margin constraint in the form of a table:

$$\frac{GM_l - GM_{l,req}}{LGMFACT} \leq 0$$

Format and Example:

1	2	3	4	5	6	7	8	9	10
DCONLGM	SID	VTYPE	LGMFACT	V1	LGM1	V2	LGM2		CONT
CONT	V3	LGM3	V4	LGM4	-etc-				

1	2	3	4	5	6	7	8	9	10
DCONLGM	1		1.0	0.0	-6.0	12060.	-6.0		CONT
CONT	12070.	0.0	50000.	0.0					

Field	Contents
SID	Constraint set identification, the constraint are referenced by the design constraint ID in Solution Control (Integer > 0)
VTYPE	Nature of the velocity referred in the table. Either TRUE for true velocity or EQUIV for equivalent air speed. Default = TRUE
LGMFACT	Constraint scaling factor (Real > 0.0, Default=1.0)
VI	Velocity value (Real ≥ 0.0)
LGMI	Required lower gain margin value (in dB) (Real ≤ 0)

Remarks:

1. Lower gain margin constraints are selected in Solution Control with the discipline option: DCON=SID.

2. The VI must be in either ascending or descending order.

3. At least two pairs must be entered.

Input Data Entry: **DCONLPM**

Description: Defines an lower phase margin constraint in the form of a table:

$$\frac{PM_l - PM_{l,req}}{LPMFACT} \leq 0$$

Format and Example:

1	2	3	4	5	6	7	8	9	10
DCONLPM	SID	VTYPE	LPMFACT	V1	LPM1	V2	LPM2		CONT
CONT	V3	LPM3	V4	LPM4	-etc-				

1	2	3	4	5	6	7	8	9	10
DCONLPM	1		1.0	0.0	-45.0	12060.	-45.0		CONT
CONT	12070.	0.0	50000.	0.0					

Field	Contents
SID	Constraint set identification, the constraint are referenced by the design constraint ID in Solution Control (Integer > 0)
VTYPE	Nature of the velocity referred in the table. Either TRUE for true velocity or EQUIV for equivalent air speed. Default = TRUE
LPMFACT	Constraint scaling factor (Real > 0.0, Default=1.0)
VI	Velocity value (Real $\geq$ 0.0)
LPMI	Required lower phase margin value (in degrees) (-180.0 $\leq$ Real $\leq$ 0)

Remarks:

1. Lower phase margin constraints are selected in Solution Control with the discipline option: DCON=SID.

2. The VI must be in either ascending or descending order.

3. At least two pairs must be entered.

Input Data Entry: **DCONUGM**

Description: Defines an upper gain margin constraint in the form of a table:

$$\frac{GM_{u,req} - GM_u}{UGMFACT} \leq 0$$

Format and Example:

1	2	3	4	5	6	7	8	9	10
DCONUGM	SID	VTYPE	UGMFACT	V1	UGM1	V2	UGM2		CONT
CONT	V3	UGM3	V4	UGM4	-etc-				

1	2	3	4	5	6	7	8	9	10
DCONUGM	1		1.0	0.0	6.0	12060.	6.0		CONT
CONT	12070.	0.0	50000.	0.0					

Field	Contents
SID	Constraint set identification, the constraint are referenced by the design constraint ID in Solution Control (Integer > 0)
VTYPE	Nature of the velocity referred in the table. Either TRUE for true velocity or EQUIV for equivalent air speed. Default = TRUE
UGMFACT	Constraint scaling factor (Real > 0.0, Default=1.0)
VI	Velocity value (Real ≥ 0.0)
UGMI	Required upper gain margin value (in dB) (Real ≥ 0)

Remarks:

1. Upper gain margin constraints are selected in Solution Control with the discipline option: DCON=SID.

2. The VI must be in either ascending or descending order.

3. At least two pairs must be entered.

Input Data Entry: **DCONUPM**

Description: Defines an upper phase margin constraint in the form of a table:

$$\frac{PM_{u,req} - PM_u}{UPMFACT} \le 0$$

Format and Example:

1	2	3	4	5	6	7	8	9	10
DCONUPM	SID	VTYPE	UPMFACT	V1	UPM1	V2	UPM2		CONT
CONT	V3	UPM3	V4	UPM4	-etc-				

1	2	3	4	5	6	7	8	9	10
DCONUPM	1		1.0	0.0	45.0	12060.	45.0		CONT
CONT	12070.	0.0	50000.	0.0					

Field	Contents
SID	Constraint set identification, the constraint are referenced by the design constraint ID in Solution Control (Integer > 0)
VTYPE	Nature of the velocity referred in the table. Either TRUE for true velocity or EQUIV for equivalent air speed. Default = TRUE
UPMFACT	Constraint scaling factor (Real > 0.0, Default=1.0)
VI	Velocity value (Real ≥ 0.0)
UPMI	Required upper phase margin value (in degrees) (0 ≤ Real ≤ 180)

Remarks:

1. Lower gain margin constraints are selected in Solution Control with the discipline option: DCON=SID.

2. The VI must be in either ascending or descending order.

3. At least two pairs must be entered.

Input Data Entry: **DINIT** Initial [D] for aero approximation

Description: The [D] matrix at the beginning of the [D] → [E] →[D] iterations in the nonlinear least-square process to obtain the minimum-state approximation coefficients.

Format and Example:

1	2	3	4	5	6	7	8	9	10
DINIT	SID	D11	D12	-etc-	D(1,nL)	D21	D22	-etc-	CONT
CONT	D(nh,nL)								

DINIT	23	1.5	-3.4	5.	7.8	-22.	1.	3.2	+AB
+AB	2.0								

Field	Contents
SID	Set identification number (Integer > 0)
Dij	Elements of the [D] matrix, column by column

Remarks:

1. SID is selected by the MINSTAT data entry.
2. The order of [D] is n_h x n_L where n_h is the number of structural modes and n_L is the number of aerodynamic roots.

Input Data Entry: **GAINSET** Set of gains

Description: Defines the set of gains of an ASE case.

Format and Example:

1	2	3	4	5	6	7	8	9	10
GAINSET	SID	GAIN1	GAIN2	GAIN3	GAIN4	GAIN5	GAIN6	GAIN7	CONT
CONT	GAIN8	-etc-							

GAINSET	10	4	2	5	1	3			

Field	Contents
SID	Set identification number (Integer > 0)
GAINi	Identification numbers of an ASEGAIN entry defining the gains of the control system

Remarks:

1. SID is selected in the ASECONT data entry.

Input Data Entry: **LMODSET** Set of load modes

Description: Defines the set of load modes to be used in gust response analysis.

Format and Example:

1	2	3	4	5	6	7	8	9	10
LMODSET	SID	LID1	LID2	LID3	LID4	LID5	LID6	LID7	CONT
CONT	LID8	-etc-							

LMODSET	70	10	20						

Field	Contents
SID	Set identification number (Integer > 0)
LIDi	Identification numbers of LOADMOD entry defining a load mode

Remarks:

1. SID is selected by the CONGUST data entry.

Input Data Entry: **MIMOSS** MIMO control element

Description: Defines a MIMO control element by its state space matrices.

Format and Example:

1	2	3	4	5	6	7	8	9	10
MIMOSS	ID	NTF	NU	NY	LMIMO				

MIMOSS	50	6	7	4	MATRC4				

Field	Contents
ID	Identification number (Integer ≥ 0)
NTF	Order of the controller (Integer ≥ 1)
NU	Number of inputs (Integer ≥ 1)
NY	Number of outputs (Integer ≥ 1)
LMIMO	Label of the DMI entry defining the A_c, B_c, C_c, D_c matrices merged in $\begin{bmatrix} A_c & B_c \\ C_c & D_c \end{bmatrix}$

Remarks:

1. All ID numbers of ACTU, ASESNSR, CJUNCT, CRESP, MIMOSS, SISOTF entries must be distinct.

2. ID is selected by the TFSET data entry.

3. The matrix defined on the DMI-LMIMO data entry must be with dimensions (NTF+NY) $\times$ (NTF+NU).

Input Data Entry: **MINSTAT** Parameters for aerodynamic approximation

Description: Defines parameters for rational approximation of unsteady aerodynamic forces by minimum-state method.

Format and Example:

1	2	3	4	5	6	7	8	9	10
MINSTAT	SID	LAGID	ITMAX	APCID	APWID	DMATID	IRED		

MINSTAT	20	10	100	21	22	23			

Field	Contents
SID	Set identification number (Integer > 0)
LAGID	Identification number of a AEROLAG set specifying approximation roots (Integer > 0)
ITMAX	Number of $[D] \rightarrow [E] \rightarrow [D]$ iterations (Integer ≥ 0)
APCID	Identification number of a APCONST set specifying approximation constraints (Integer > 0, or blank)
APWID	Identification number of a PWEIGHT set specifying parameters for weighting of the approximation data (Integer > 0, or blank)
DMATID	Identification number of a DINIT set specifying the initial [D] in the iterative approximation process (Integer ≥ 0, or blank)
IRED	Number of highest frequency modes that are candidates for dynamic reduction (Integer ≥ 0, or blank)

Remarks:

1. SID is selected by the ASESOL data entry.

2. if LAGID is zero, aerodynamic approximation is performed with no lag terms ($n_L = 0$).

3. if ITMAX is zero, the Roger's approximation is performed.

4. if APCID is blank or zero, no approximation constraints are applied.

5. if APWID is blank or zero, the weighting reflects the normalization of the structural modes to unit generalized masses.

6. if DMATID is blank or zero, or ITMAX = 0, the initial [*D*] is build by unit matrices whose order is the smaller dimension of [*D*].

Input Data Entry: **PWEIGHT** Aero approximation weighting parameters

Description: Defines physical-weighting parameters for rational approximation of unsteady aerodynamic forces by minimum state technique.

Format and Example:

1	2	3	4	5	6	7	8	9	10
PWEIGHT	SID	PRO	PV	WCUT	NWD	PDAMP	PCONT	LGP	CONT
CONT	GGID	GC							

PWEIGHT	20	1.147E-7	33358.3	0.01	2	0.0			

Field	Contents
IDP	ID number that refers to FLCOND from discipline FLUTTER (Integer > 0)
PRO	Air density at a selected design point for physical weighting (Real)
PV	True air speed at the design point (Real)
WCUT	Minimal maximum absolute value of weighted aerodynamic term (Real)
NWD	Number of weight peak widening cycles (Integer)
PDAMP	Dimensionless modal damping (Real)
PCONT	Set identification number of the ASECONT entry which defines the control system for physical weighting (Integer > 0 or blank)
LGP	Scale of turbulence for gust physical weighting (Real > 0 or blank)
GGID	ID number of grid or scalar point at which acceleration response is considered for gust physical weighting (Integer > 0 or blank)
GC	Component number of GGID for gust response (0÷6 or blank)

Remarks:

1. SID is selected by the MINSTAT data entry.

2. The density, velocity and damping specified in this data entry are only for the purpose of physical weighting of the aerodynamic data in the rational approximation process.

3. The default value for PCONT is CONTID of the parent ASESOL data entry.

4. If physical weighting is requested for gust columns, LGP, GGID, and GC can not be blank.

Input Data Entry: **RESPSET** Set of gust response points

Description: Defines the set of gust response points.

Format and Example:

1	2	3	4	5	6	7	8	9	10
RESPSET	SID	RESP1	RESP2	RESP3	RESP4	RESP5	RESP6	RESP7	CONT
CONT	RESP8	-etc-							

RESPSET	20	7	3	5	2	3			

Field	Contents
SID	Set identification number (Integer > 0)
RESPi	Identification numbers of CRESP entry defining a gust response point

Remarks:

1. SID is selected in the CONGUST data entry.

Input Data Entry: **SENSET** Set of sensors

Description: Defines the set of sensors of an ASE case.

Format and Example:

1	2	3	4	5	6	7	8	9	10
SENSET	SID	SENS1	SENS2	SENS3	SENS4	SENS5	SENS6	SENS7	CONT
CONT	SENS8	-etc-							

SENSET	20	7	3	5	2	3			

Field	Contents
SID	Set identification number (Integer > 0)
SENSi	Identification numbers of an ASESNSR entry defining a sensor

Remarks:

1. SID is selected in the ASECONT data entry.

Input Data Entry: **SISOTF** SISO control element

Description: Defines a SISO controller by a transfer function.

Format and Example:

1	2	3	4	5	6	7	8	9	10
SISOTF	ID	NDEN	NNUM	A0	A1	A2	A3	A4	CONT
CONT	A5	-etc-	A(NDEN-1)	B0	B1	B2	B3	B4	CONT
CONT	B5	-etc-	B(NNUM)						

SISOTF	70	3	1	0.3	0.2	0.1	0.15	0.05	

Field	Contents
ID	Identification number (Integer ≥ 0)
NDEN	Order of the denominator (Integer ≥ 0)
NNUM	Order of the numerator (NDEN $\geq$ Integer ≥ 0)
Ai	Coefficients of the denominator polynomial (Real)
Bi	Coefficients of the numerator polynomial (Real)

Remarks:

1. All ID numbers of ACTU, ASESNSR, CJUNCT, CRESP, MIMOSS, SISOTF entries must be distinct.

2. ID is selected in the TFSET data entry.

3. If NDEN = 0, the other entries are ignored. A zero-order controller is defined with a unit gain.

4. The transfer function is:

$$TF = \frac{B(NNUM)* s^{NNUM} + \cdots + B0}{s^{NDEN} + A(NDEN-1)* s^{NDEN-1} + \cdots + A0}$$

Input Data Entry: **SURFSET** Control surfaces

Description: Defines the set of control surfaces of an ASE case.

Format and Example:

1	2	3	4	5	6	7	8	9	10
SURFSET	SID	SURF1	SURF2	SURF3	SURF4	SURF5	SURF6	SURF7	CONT
CONT	SURF8	-etc-							

SURFSET	30	SURF2	SURF4	SURF6	SURF3	SURF1			

Field	Contents
SID	Set identification number (Integer > 0)
SURFi	The name of the *i-th* control surface for ASE analysis (Character)

Remarks:

1. SID is selected in the ASECONT data entry.

2. The SURFi labels must be defined by the AESURF data entries.

Input Data Entry: **TFSET** Control element set

Description: Selects the set of control elements of an ASE case.

Format and Example:

1	2	3	4	5	6	7	8	9	10
TFSET	SID	TF1	TF2	TF3	TF4	TF5	TF6	TF7	CONT
CONT	TF8	-etc-							

TFSET	40	2	3	5	4	1			

Field	Contents
SID	Set identification number (Integer > 0)
Tfi	Distinct identification numbers of CJUNCT, MIMOSS, SISOTF entries (Integer > 0)

Remarks:

1. SID is selected in the ASECONT data entry.

Input Data Entry: **VINIG** New set of design variables

Description: Defines an initial set of design variables for analysis or optimization which are based on OLD data base.

Format and Example:

1	2	3	4	5	6	7	8	9	10
VINIG	DVID	VMIN	VMAX	VALUE	LABEL				

VINIG	103	0.01	2.0	0.45	ZONE8				

Field	Contents
DVID	Design variable identification (Integer > 0)
VMIN	Minimum allowable value for the design variable (Real $\geq$ 0)
VMAX	Maximum allowable value for the design variable (Real $\geq$ 0)
VALUE	Initial value of the design variable (Real, VMIN $\leq$ VALUE $\leq$ VMAX)
LABEL	Optional user supplied label to define the design variable (Text)

Remarks:

1. The VINIG entry is needed only if one wants to change the design values with which the data base was created.